Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное
учреждениевысшего профессионального образования
«Уральский государственный педагогический университет»
Институт математики, информатики и информационных технологий
Кафедра информатики, информационных технологий и методики обучения информатике

МОДЕЛИРОВАНИЕ КРУПНОМАСШТАБНЫХ ЭКОЛОГИЧЕСКИХ СИСТЕМ НА ОСНОВЕ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ

Выпускная квалификационная работа
бакалавра по направлению подготовки
02.03.02 - *Фундаментальная информатика и информационные технологии*

Исполнитель: студент группы Б-41
Института математики, информатики и ИТ
Солдатова Е. В.
Руководитель: к.ф.-м.н., профессор кафедры ИИТиМОИ
Подчиненов И. Е.

Работа допущена к защите
«____»_____ 201__ г.
Зав. кафедрой _____

Екатеринбург 2017.

ОГЛАВЛЕНИЕ

РЕФЕРАТ

Солдатова Е. В. .МОДЕЛИРОВАНИЕ КРУПНОМАСШТАБНЫХ ЭКОЛОГИЧЕСКИХ СИСТЕМ НА ОСНОВЕ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ, выпускная квалификационная работа: 38 стр., рис. 17, табл. 2, библ. 30 назв., приложений 1.

Ключевыеслова: загрязнение водоема, конечные разности, полиномиальная аппроксимация, параллельные вычисления.

Объектразработки –моделирование сложных экологических систем.

Цельработы – разработать алгоритм и реализовать компьютерную программу, которая позволит моделировать загрязнение озера каким-либо веществомидеструктуризацию этого вещества в любой момент времени.

В работе показана возможность использования графического процессора для моделирования распространения природного загрязнения водоемов. Реализован алгоритм, с использованием технологий OpenMP (Open Multi-Processing — открытый стандарт для распараллеливания программ на языках Си, Си++ и Фортран) и CUDA (англ. ComputeUnifiedDeviceArchitecture — программно-аппаратная архитектура параллельных вычислений), демонстрирующий сравнение моделирования на центральном и графическом процессорах.

Введение

В настоящее время, в связи с быстрым экономическим ростом, и увеличением потребления ресурсов возрастает актуальность решения проблемы загрязнения окружающей среды, в частности водоемов, которые чаще всего являются источниками пресной воды. Важность пресной воды трудно переоценить – она необходима для жизнедеятельности всех организмов, и, конечно же, для человека. Человек широко использует поверхностные воды, однако, после использования вода возвращается в водоемы в меньшем количестве и в худшем качестве [10].

Существуют такие источники загрязнения как:

- химические: отходы производств, удобрения, вымываемые из почвы;
- биологические: бытовые сточные воды, отходы сельхозпроизводства, загрязнение от курортных зон;
- тепловые и радиационные - стоки от ТЭС и АЭС;
- физические- смыв песка, пыли.

Таким образом, возрастает угроза недостатка чистой воды [25].

Для прогнозирования последствий загрязнения, устранения этих загрязнений наиболее оптимальным способом и оценки их влияния на биологические системы и экономику широко используется компьютерное моделирование. С помощью компьютерных моделей можно изучать динамику распространения загрязнений, а также осуществлять контроль над ними [26].Математическая модель такой задачи сводится к дифференциальным уравнениям в частных производных, а ее решение даже на двумерной сетке требует выполнения большого числа операций, поэтому в работе выполнены расчеты с использованием параллельных вычислений.

Если реализовать такиекомпьютерные модели, применяя параллельные алгоритмы, то это позволит использовать все возможности современной компьютерной техники для того, чтобы моделировать поведение системы в течение длительного временного интервала за секунды. Таким образом, можно рассчитывать различные варианты развития событий.

В настоящее время параллельные вычисления – актуальная тема для исследования. Производительность компьютерных систем непрерывно повышается. Росту технологий способствует потребность решения сложных вычислительных задач за короткое время. Использование видеокарты GPU, позволяет производить большое количество однотипных вычислений. То есть, с решением простейших независимых математических задач, GPU справится намного быстрее, чем центральный процессор CPU.

В ходе работы был проанализирован ряд источников по данной теме.

В статье Оби Эммануэль Оду «Математическое моделирование нефтяного загрязнения прибрежных вод республики Нигерия» [27] описывается построение математической модели загрязнения озера с использованием программного комплекса «Slickmovement».

В статье А. В Игнатова и В. В. Кравченко «Информационное моделирование загрязнения водных объектов» [28] построено несколько моделей для оценки динамики загрязнения водоемов, решены практические задачи и приведены прогностические расчёты.

В статье В.И. Климук и А.В.Дударева «Математическое моделирование динамики пассивной примеси в центральной части озера Селигер» [29] выполнены численные эксперименты и расчёты, для которых использовались среднемесячные поля осредненных по глубине ветровых течений, найденные из решения системы уравнений.

Объект исследования–моделирование сложных экологических систем.

Предмет исследования –создание модели загрязнения водоема с использованием параллельных вычислений.

Цель данной работы–разработать алгоритм и реализовать компьютерную программу, которая позволит моделировать загрязнение озера каким-либо веществомидеструктуризацию этого вещества в любой момент времени.

Для достижения данной цели необходимо решить следующие задачи:

- сформулировать математическую модель распространения загрязнения в водоеме;

- построить конечно-разностный алгоритм решения двумерного уравнения распространения загрязнений в водоеме;

- реализовать данный алгоритм на многоядерной системе с использованием технологии OpenMP;

- провести исследованиекомпьютерной модели на предмет эффективности параллельных алгоритмов в зависимости от параметров (время, коэффициент Шези, скорость течения, средняя глубина водного объекта, коэффициент скорости поглощения частиц).

Глава 1. Математическая модель загрязнения водоема

1.1. Система уравнений

В данной работе диффузия, растекание, и испарение моделируются с использованием двумерного уравнения распространения загрязнений

$$\frac{\partial C}{\partial t} + \frac{\partial C}{\partial x} + v\frac{\partial C}{\partial y} - K_{ix}\frac{x^2 C}{dx^2} - K_{iy}\frac{d^2 C}{dy^2} + f(C) = 0 \qquad (1)$$

где C– концентрация, *кг/м*^{2};

t – время;

u,v – компоненты вектора скорости течения, в общем случае зависящие от времени, *м/с*^{2};

x,y – декартовы координаты;

K_x, K_y – коэффициенты турбулентной диффузии вдоль осей координат x и y соответственно, *м*^{2}*/с*. Расчёт этих коэффициентов рассмотрен в параграфе 1.2.

Функция f(C) описывает соответствующий процесс деструкции и в зависимости от характера процесса деструкции имеет разный вид.

Краевые условия для уравнения, которые будем определять так:

$$f(C) = \alpha C = \alpha C, \qquad (2)$$

где α – скорость поглощения частиц.

1.2. Метод конечных разностей. Полиномиальная аппроксимация

Для численного решения данной задачи был использован метод конечных разностей.

Метод конечных разностей (метод сеток) является методом численного решения краевых задач для дифференциальных уравнений.

Для того чтоб построить численное решение необходимо составить разностную схему, которая приближенно описывает дифференциальное уравнение. Для этого область непрерывного изменения аргумента функции

заменяют областью его дискретного изменения. Таким образом, строится сетка, и рассматриваются сеточные функции – функции, которые определенны только в узлах этой сетки. Граничные условия и производные, которые входят в дифференциальные уравнения приближенно заменяются их разностными аналогами (линейными комбинациями значений сеточных функций в узлах сетки) и выражаются через неизвестные узловые значения искомой функции[12]. То есть, граничную задачу заменяют дискретной краевой задачей или разностной схемой, которая представляет собой систему линейных алгебраических уравнений. За приближенное решение рассматриваемой задачи принимают решение, полученное с помощью этой разностной схемы [11].

Популярность данного метода можно объяснить простотой подхода к дискретизации дифференциальных уравнений, но следует учитывать, что для одной граничной задачи можно построить большое число разностных схем, и не все из них будут приводить к искомому результату [11]. Этот метод является универсальным, хотя и дает приближенную информацию о решении.

Используем метод полиномиальной аппроксимации.

Метод полиноминальной аппроксимации основан на применении непрерывной функции со свободными параметрами. Непрерывную функцию в определенном интервале можно аппроксимировать полиномом высокого порядка. Другими словами, применяется аппроксимирующая функция, заменяющая исходную функцию, которая определяется по значениям в узлах сетки. Эта аппроксимирующая функция впоследствии аналитически дифференцируется. Обычно в качестве таких функций используют полиномы. Степень таких полиномов относительно не высока и должна быть связана с порядком дифференциального уравнения. Если найден полином, который достаточно точно аппроксимирует функцию, то координаты точки оптимума функции можно оценить путем вычисления координаты точки оптимума полинома [13].

Это распространенный метод нахождения производных по экспериментальным данным. Обычно такие представления удобны для относительно гладких функций и для задания граничных условий.

С увеличением порядка аппроксимации полинома аппроксимирующие функции становятся чувствительными к погрешностям вычислений, которые неминуемы в приближенных вычислениях. Это приведет к неправильному результату при вычислении значения функции [1].

Рассмотрим полином второго порядка:

$$C(x,y,t) = a + bx + cx^2 + gy + my^2 + pt + qt^2. \quad (3)$$

Будем считать, что разложение в точке *i*соответствует началу координат x=0. Использоваться будут три узла i-1, j, i+1.

Составим систему:

$$\begin{cases} C_{i,j,k} = a \\ C_{i-1,j,k} = a - b\Delta x + c\Delta x^2 \\ C_{i+1,j,k} = a + b\Delta x + c\Delta x^2 \\ C_{i,j-1,k} = a - g\Delta y + m\Delta y^2 \\ C_{i,j-1,k} = a + g\Delta y + m\Delta y^2 \\ C_{i,j,k-1} = a - q\Delta t + p\Delta t^2 \\ C_{i,j,k-1} = a + q\Delta t + p\Delta t^2 \end{cases} \quad (4)$$

Решая систему, получим такие коэффициенты уравнения:

$$b = \frac{C_{i+1,j,k} - C_{i-1,j,k}}{2\Delta x}, \quad (5)$$

$$c = \frac{C_{i+1,j,k} + C_{i-1,j,k} - 2C_{i,j,k}}{2\Delta x^2}, \quad (6)$$

$$g = \frac{C_{i,j+1,k} - C_{i,j-1,k}}{2\Delta y}, \quad (7)$$

$$m = \frac{C_{i,j+1,k} + C_{i,j-1,k} - 2C_{i,j,k}}{2\Delta y^2}, \quad (8)$$

$$q = \frac{C_{i,j,k+1} - C_{i,j,k-1}}{2\Delta t}, \quad (9)$$

$$p = \frac{C_{i,j,k+1} + C_{i,j,k-1} - 2C_{i,j,k}}{2\Delta t^2}. \quad (10)$$

Отсюда значения производных примут вид:

$$\frac{dC}{dx} = (b + 2cx)_{x=0} = b, \quad (11)$$

$$\frac{d^2C}{dx^2} = 2c, \quad (12)$$

$$\frac{dC}{dy} = (g + 2my)_{y=0} = g, \quad (13)$$

$$\frac{d^2C}{dy^2} = 2m, \quad (14)$$

$$\frac{dC}{dt} = (q + 2pt)_{t=0} = q, \quad (15)$$

Получили численное решение уравнения 1:

$$\frac{C_{i,j,k+1} - C_{i,j,k-1}}{2\Delta t} + u\frac{C_{i+1,j,k} - C_{i-1,j,k}}{2\Delta x} + v\frac{C_{i,j+1,k} - C_{i,j-1,k}}{2\Delta y} -$$
$$K_{lx}2\frac{C_{i+1,j,k} + C_{i-1,j,k} - 2C_{i,j,k}}{2\Delta x^2} - K_{ly}2\frac{C_{i,j+1,k} + C_{i,j-1,k} - 2C_{i,j,k}}{2\Delta y^2} + f(C) = 0 \quad (16)$$

$$C_{i,j,k-1} - u\frac{C_{i+1,j,k} - C_{i-1,j,k}}{\Delta x}\Delta t - v\frac{C_{i,j+1,k} - C_{i,j-1,k}}{\Delta y}\Delta t + K_{lx}\frac{C_{i+1,j,k} + C_{i-1,j,k} - 2C_{i,j,k}}{\Delta x^2}2\Delta t$$
$$+K_{ly}\frac{C_{i,j+1,k} + C_{i,j-1,k} - 2C_{i,j,k}}{\Delta y^2}2\Delta t - f(C)2\Delta t = C_{i,j,k+1} \quad (17)$$

1.3. Расчет коэффициентов турбулентной диффузии. Коэффициент Шези.

Рассмотрим понятие турбулентной диффузии.

Под турбулентной диффузией понимают перенос вещества в пространстве, обусловленный турбулентным движением среды.

Под турбулентным движением понимается вихревое движение, при котором, частицы газа или жидкости случайным образом двигаются по сложным траекториям, с переменной скоростью. Таким образом, если определенное множество частиц среды находятся рядом, то в последующем они рассеиваются по пространству, и с течением времени расстояние между ними возрастает [7].

Коэффициентдиффузии - это количественная характеристика скорости диффузии. Коэффициентпропорциональности,представляющий количество вещества, диффундирующегосквозьединицуплощадиприединичномградиентеконцентрации заединицу времени [8].

Существует ряд формул, предназначенных для определения коэффициента турбулентной диффузии. Например, формула Маккавеева:

$$D = \frac{g H_{ср} V_р}{37 n_{ш} C^2} \qquad (18)$$

где g = 9,81 м^2/с – ускорение свободного падения;

$H_{ср}$ – средняя глубина водного объекта на расчетном участке, м;

$V_р$ – скорость течения реки, м/с;

$n_ш$ – коэффициент шероховатости ложа водного объекта (прилож. 1);

С – коэффициент Шези [2].

Коэффициент Шези – это коэффициент сопротивления трения по длине, интегральная характеристика сил сопротивления. Величина коэффициента Шези характеризует энергические потери в движущихся жидкостях.

Под гидродинамическим сопротивлением понимают сопротивление движению жидкости, вызванное соприкосновением на границе потока с другими телами [10].

Коэффициент Шези определен по формуле Павловского:

$$C = \frac{H_{ср}^{1.6}}{n_{ш}} \quad (19)$$

где $H_{ср}$ – средняя глубина водного объекта на расчетном участке, м;

$n_{ш}$ – коэффициент шероховатости ложа водного объекта.

C – коэффициент Шези [14].

Отдельно рассмотрим коэффициент шероховатости ложа водного объекта. Как известно, на дне водоема накапливаются осадки в виде взвеси, которые постепенно оседают, увеличивая шероховатость дна и сопротивление потоку жидкости. Это, а так же большое количество других факторов, привело к необходимости введения некоего коэффициента. Был предложен коэффициент шероховатости, который определялся по описательной таблице. Изменение коэффициента шероховатости заметно влияет на результаты расчета, поэтому важно определить сложившийся на практике коэффициент шероховатости ложа водного объекта. Для определения коэффициента шероховатости созданы специальные таблицы, в которых приведены значения коэффициентов и соответствующие им описание русла водных объектов [9]. Широкое распространение получила таблица Скрибного (см. приложение).

Глава2. Аппаратные и программные средства для построения модели загрязнения

2.1. Архитектура современных процессоров

В середине прошлого века в связи с развитием таких областей как энергетика, разработка сложных технических устройств, планирование, ракетостроение и т.д. остро возникла необходимость обработки больших объемов данных. Эффективная работа в этих сферах невозможна без использования компьютера. Производительность компьютерных систем непрерывно повышается. Росту технологий способствует потребность решения сложных вычислительных задач в короткое время. Соответственно совершенствуются и средства создания компьютерной техники.

Одна из формулировок закона Мура гласит, что доступная вычислительная мощность удваивается каждые 18 месяцев [15]. До недавнего времени закон Мура достигался за счёт уменьшения расстояния между электродами транзисторов на кристалле. При этом энергопотребление и тепловыделение транзисторов уменьшается, а значит, появляется возможность размещения большего числа транзисторов. Таким образом, основной способ повышения производительности – рост тактовой частоты. Но так как увеличение тактовой частоты приводит к росту потребляемой мощности и тепловыделению, и имеет другие технологические ограничения, со временем повышение производительности за счёт тактовой частоты стало невозможным [21]. На сегодняшний день многие пути совершенствования процессоров в этом направлении оказались исчерпаны. Альтернативный путь - масштабирование в ширину, т.е. увеличение числа ядер на единственном кристалле. Оказалось возможным использовать менее энергоемкие и дорогие процессоры, более простые в изготовлении и более надежные. Снижается сложность логики процессоров. Максимальная производительность в этом случае равна сумме

производительности вычислительных ядер. Исходя из вышесказанного, многоядерность обладает рядом существенныхплюсов и поэтому является наиболее перспективным направлением в данный момент [16].

Кроме того, ускоряет решение задач параллельное выполнение множества программ для различных значений параметров. Нужно отметить, что для обслуживания вычислительных потребностей большой группы пользователей эффективнее использование суперкомпьютера, чем эквивалентного количества однопроцессорных рабочих станций, в силу того, что при помощи системы управления заданиями легче обеспечить равномерную нагрузку вычислительных ресурсов [22].

Как правило, операционные системы суперкомпьютеров разделяют ресурсы одного процессора между несколькими программами, которые выполняются одновременно. Это делается для достижения максимальной скорости выполнения программ. Поэтому, как два противоположных варианта, возможны следующие режимы использования n-процессорной системы:

- все ресурсы отдаются для выполнения одной программы, при этом ожидаемо n-кратное ускорение работы программы по сравнению с однопроцессорной системой;
- одновременно выполняется n обычных однопроцессорных программ, при этом, на скорость выполнения программы пользователя другие программы не будут оказывать влияния [19].

Одной из самых известных и ранних классификаций архитектур компьютера является классификация М. Флинна. В основе этой классификации лежит понятие потока. Поток – это последовательность данных или команд, обрабатываемых процессором. По числу потоков команд и потоков данных существуют следующие классы архитектур: MIMD, SIMD,SISD,MISD[20].

SISD (SingleInstructionSingle Data)– архитектура параллельных вычислений в которой одна инструкция, выполняется над одним набором

данных. Например, за один такт суммируются два числа, то есть выполняется одна операция сложения.

SIMD (SingleInstructionMultipleData)–архитектура параллельных вычислений, в которой одна и та же команда выполняетсяодновременно для нескольких наборов данных, например, операция сложения, которая за одну инструкцию складывает сразу 8 чисел за 1 такт. Компьютер с данной архитектурой может состоять, как из большого количества простых процессоров (GPU), так и из длинных регистров (инструкции SSE в x86 совместимых процессорах).

Такая архитектура обеспечивает высокую производительность в узком круге задач, например, сложение векторов, матриц.

MISD (MultipleInstructionSingle Data)– архитектура параллельных вычислений в которой, выполняются различные наборы команд, и один набор данных. В этой архитектуре данные подаются на набор процессоров, каждый из которых исполняет свою программу их обработки.

MIMD (MultipleInstructionsMultipleData)– архитектура параллельных вычислений, в которой различные наборы команд независимо выполняются для различных наборов данных. Данная архитектура делится на 2 типа:

- **системы с распределённой памятью –** данную систему легко продемонстрировать на примере вычислительного кластера. У каждого узла кластера есть своя память, но другой узел не может напрямую получить доступ к памяти другого узла. Каждый такой узел обменивается сообщениями по сети, которая соединяет все узлы. Эти сообщения используются для связи между узлами, т.е. для записи, чтения и удаления блоков памяти. Примером такой технологии может служить MPI.
- **Системы с совместно используемой памятью**– все процессоры (или ядра) обращаются к одной общей памяти через общую шину

данных. На данный момент все современные многоядерные компьютеры, будь то домашний компьютер, или телефон, являются такими. У них есть многоядерный процессор, которой на каждом своем ядре может одновременно выполнять разные программы и общая память называется ОЗУ или RAM[18].

2.3. OpenMP и параллельное программирование

Развитие области высокопроизводительных вычислений направленона уменьшение времени выполнения последовательной программы и на проблему нехватки оперативной памяти из-за большого объема данных. В этих случаях применяется параллельное программирование.

Интерфейс OpenMP – одна из наиболее популярных технологий параллельного программирования. OpenMP успешно используется, как при программировании суперкомпьютерных систем с большим количеством процессоров, так и в настольных пользовательских системах.Первый стандарт OpenMP был разработан в 1997 г. как API, ориентированный на написание легко переносимых многопоточных приложенийнамногопроцессорных системахс общей памятью. Сначала он был основан на языке Fortran, но позднее включил в себя и языки Си и Си++ [6].

Каждая программа имеет один главный поток выполнения - набор инструкций, которые выполняются последовательно одна за другой. В этом случае задействовано только одно ядро процессора. Иногда необходимо сделать так, чтобы некоторые инструкции выполнялись не последовательно, а одновременно - в параллельном режиме. В этом случае производительность не растает линейно с увеличением количества ядер. То есть задействование четырёх ядер не гарантирует увеличение производительности в четыре раза. Тем не менее, прирост производительности происходит [5].

Работа OpenMP-программы начинается с основного потока, он пока единственный. При входе в параллельные регионы (которые могут содержаться в программе), основной поток создает группы потоков. Вконце параллельного региона группы потоков останавливаются [3]. Основной поток порождает дочерние потоки по мере необходимости. Эти потоки синхронизируются в конце параллельного блока кода, и мы снова возвращаемся к одному главному потоку. Взаимодействие потоков происходит через разделяемые переменные.

Директивы OpenMP начинаются с *#pragma omp*. Так как OpenMP контролируется*#pragma*, то код на C++ корректно скомпилируется любым компилятором C++,потому что неподдерживаемые *#pragma* должны игнорироваться. Однако OpenMP API содержит также несколько функций, и, чтобы ими воспользоваться, необходимо подключить заголовочный файл.

Директива for разделяет цикл for между текущей группой потоков, так что каждый поток в группе обрабатывает свою часть цикла [5].

```
1  extern int parallelism_enabled;
2
3    #pragma omp parallel for if(parallelism_enabled)
4
5    for(int c=0; c<n; ++c)
6
7      handle(c);
```

Рис. 1. Пример кода с использованием OpenMP

2.3. CUDA

В данной работе описанный алгоритм для нахождения концентрации вещества в водоеме, помимо реализации на центральных процессорах с использованием технологии OpenMP, так же был реализован с использованием технологии NVidiaCUDA.

CUDA – это технология компании NVidia, которая позволяет увеличить вычислительную производительность, благодаря использованию GPU.

CUDASDK является расширением языка C, которое позволяет писать программы на языке C, где некоторые части программы будут исполнятся на GPU.Это кроссплатформенное программное обеспечение.

Данная технология позволяет задействовать «видеокарты» в расчетах задач общего назначения. Под общим назначением понимается, то, что видеокарта считает не пиксель цвета на экране, а сохраняет результат в виде float, int или любой другой тип в своей памяти, а центральный процессор уже собирает результаты расчетов и производит необходимые действия дальше, например, записывает результат в файл.

На самом деле, термин «видеокарта» не совсем корректен, т.к. вычисления производятся не совсем на видеокартах, а на графических ускорителях. Графические ускорители серии Tesla, которые ставят в суперкомпьютеры, не имеют видеовыхода на монитор, они не поддерживают графическое API, такое как DirectX, OpenGL, VulcanAPI.Но видеокарты серии GeForce, которые используют обычные пользователи для обработки и вывода графической информации на монитор так же поддерживают технологию NVidiaCUDA.При вычислениях с двойной точностью их производительность падает в несколько раз, а так же они не поддерживают память с коррекцией ошибок (ECC), в отличии от серии Tesla. Конечно, можно задать вопрос: «А зачем вообще нужна память с коррекцией ошибок?». Ответ прост, когда проводятся долгие расчеты, которые постоянно пишут в память или читают из нее, в ней могут происходить ошибки, которые могут повлиять на результаты вычислений и привести к не правильному ответу, и такие случаи не так уж и редки [23].

Большинство суперкомпьютеров, состоящих в рейтинге ТОП-50, содержат графические ускорители по простой причине – графические ускорители дают лучшие соотношение производительность на ватт. Так современные процессоры потребляют приблизительно 150 Ватт и имеют

производительность около 1TFLOPS, тогда как NVidiaTeslaP100 потребляет коло 250 ватт и имеет произвольность около 5 TFLOPS. А в некоторых задачах разница может быть и больше чем в 5 раз. Из всегоэтого можно сделать вывод, что ГП это серебреная пуля, но как всегда есть одно большое но. Видеокарты, по сути, является SIMD устройствами, а это значит, что они выигрывают там, где нужно обработать параллельно большой объем данных так, чтобы мы над этими данными выполняли какие-то одни и те же инструкции одновременно, например, складывали два числа между собой, или складывание/умножение векторов, матриц. А при добавлении ветвлений производительность начинает падать т.к. выполняются сначала все нити с одним условием, а потом с другим [24]. Так же ГП имеют не такое строение оперативной памяти как RAM, которую использует ЦП. Эта память имеет большую пропускную способность в 4000 бит против 128 у процессора, но она рассчитана на извлечение большого объема данных, т.е. если извлечь один элемент, то будет извлечено еще порядка двух десятков, рядом стоящих. Тогда же, как у процессора время доступа в любую ячейку памяти одинаково изанимает на много меньше времени [17].

Технология NVidiaCUDA состоит из аппаратной части. Это видеокарты или графические ускорители, которые поддерживают данную технологию. А также набор программного обеспечения CUDASDK, которое позволяет компилировать код, который будет, исполнятся на графических чипах [30]. CUDASDK является расширением языка C++, содержащем дополнительное программное обеспечение такое, как дебагеры, профайлеры и т.д.

В CUDA есть следующие виды памяти:

- текстурная память.
- глобальная
- локальная,
- константная
- разделяемая,

- регистровая

Разделяемая память, это память общая для одного smx-устройства. Эта память позволяет обмениваться информацией внутри одного smx-устройства.

Глобальная память – общая память. Самая большая по объему общая память, которая исчисляется десятками Гб (TeslaP100 содержит 12 Гб).

Локальная память это память, которая выделяется компилятором для переменных внутри функции.

Константная память – память, в которой хранятся константы.

Текстурная память используется для работы с изображениями или текстурами.

Регистровая память является самой быстрой памятью, но прямого доступа к ней не существует. Эта память управляется компилятором [4].

Общая схема этой памяти приведена на рис. 2.

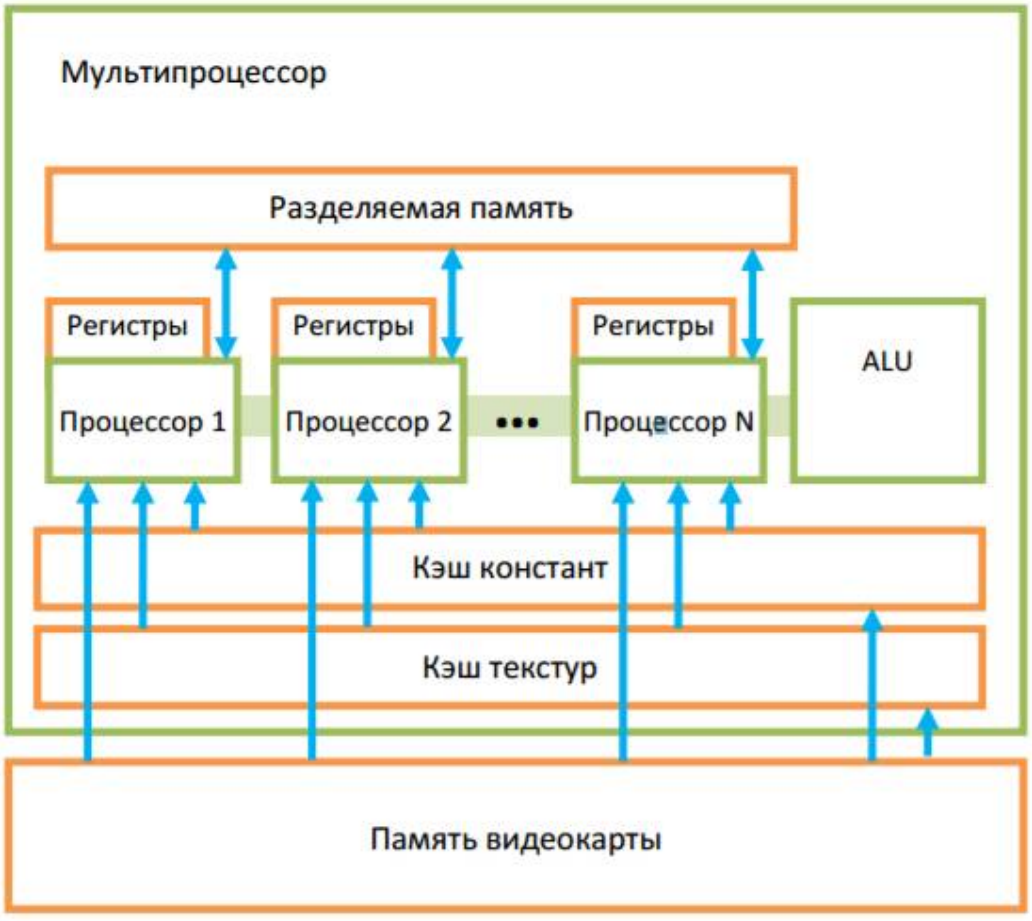

Рис 2. Модель мультипроцессора NVIDIA

2.4.Реализация алгоритма

Алгоритм был реализован на C++ с использованием технологии OpenMP и CUDA.

Для реализации алгоритма были выбраны следующие параметры: поле с сеткой размером 3500 на 3500 (соответствует водоему площадью 12,25 кв. км); шаг сетки1 м, и шаг по времени 1 с; площадь начального загрязнения

3600 кв.м; время 3600 с. То есть, смоделирована ситуация распространения

загрязнения в водоеме площадью 12,25 кв. км, в течение 3600 с

Расчет проводился на двух шестиядерных процессорах intelXeone5-2620 с тактовой частотой 2,1 Ггерц на видеокарте NVdiaTesla K20.

В таблице 1 представлены время работы ускорения и эффективности программы для процессорной версии с использованием OpenMP.

Таблица 1. Время работы, ускорения и эффективность программы.

Количество ядер	Время с.	Ускорение	эффективность
1	401	-	-
2	345	1,16231884	0,58115942
4	286	1,4020979	0,350524476
6	257	1,56031128	0,260051881
8	235	1,70638298	0,213297872
10	221	1,81447964	0,181447964
12	193	2,07772021	0,173143351

Можно заметить, что ускорение и эффективность возрастают при использовании большего количества ядер, но они возрастают непропорционально числу задействованных ядер. При увеличении числа ядер с 1 до 12, программа работает быстрее всего в 2 раза.

В таблице 2 представлено сравнение времени работы данной программы на процессоре и на видеокарте.

Таблица 2 Время работы программы на CPU и GPU

Устройство	Время в секундах
Процессор (2x intelXeone5-2620)	193,2
Видеокарта(NvidiaTesla K20)	29,9

Сравнение показало, что использование видеокарты сокращает время работы данной программы примерно в 6 раз.

Таким образом, можно сделать вывод, что для решения задач данного типане целесообразно использовать многоядерные процессоры.

На рис.3-4 изображено распространение загрязнения в зависимости от изменения скорости поглощения частиц.

Были выбраны следующие параметры: поле с сеткой размером 3500 на 3500 (соответствует водоему площадью 12,25 кв. км.); шаг сетки - 1 м, и шаг по времени 1 с.; радиус начального загрязнения – 3 600 кв.м.; время – 3600 с; средняя глубина – 10 м.; коэффициент Шези –0,03; скорость поглощения частиц – 0,1.

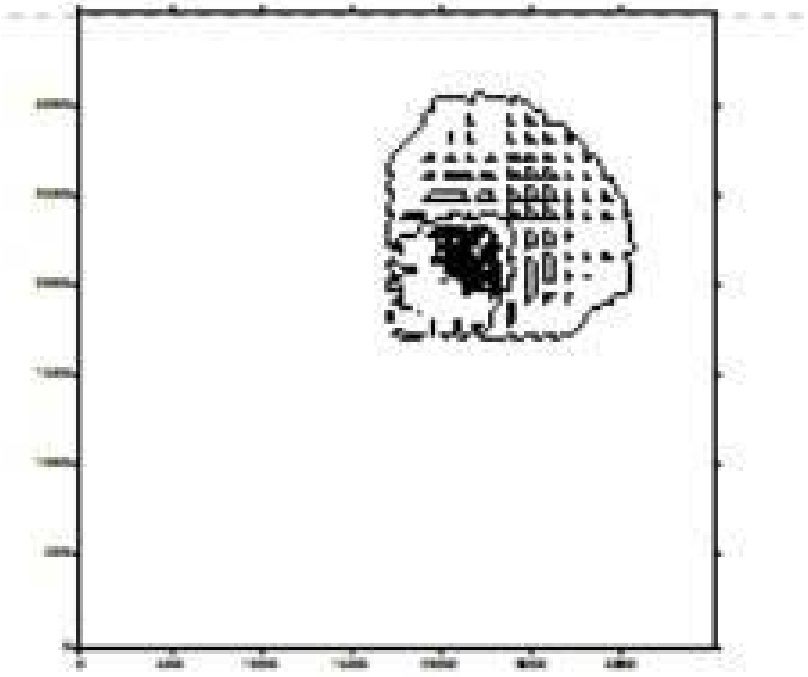

Рис 3. Изменение скорости поглощения частиц=0,99 на глубине 10 м

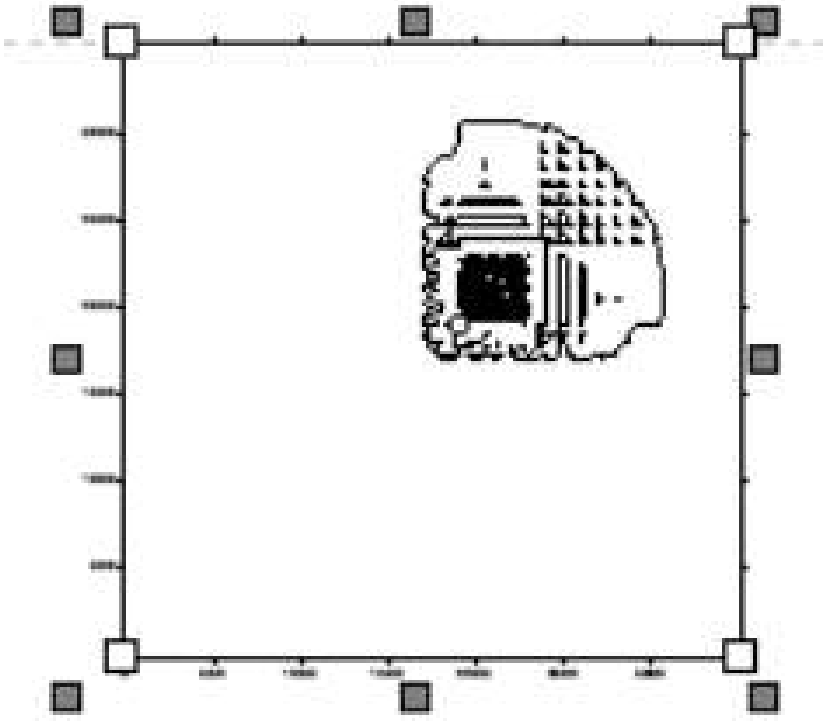

Рис.4.Изменение скорости поглощения частиц=0,1 на глубине 10 м

Таким образом, чем выше скорость поглощения частиц, тем быстрее снижается концентрация вещества в каждой точке.

На рис.5-6 изображено распространение загрязнения в зависимости от изменения коэффициента шероховатости ложа водного объекта n на глубине 10 метров. Значения коэффициента шероховатости n см. в таблице 1. Изменение коэффициента шероховатости мало влияет на распространение загрязнения на глубине 10 метров.

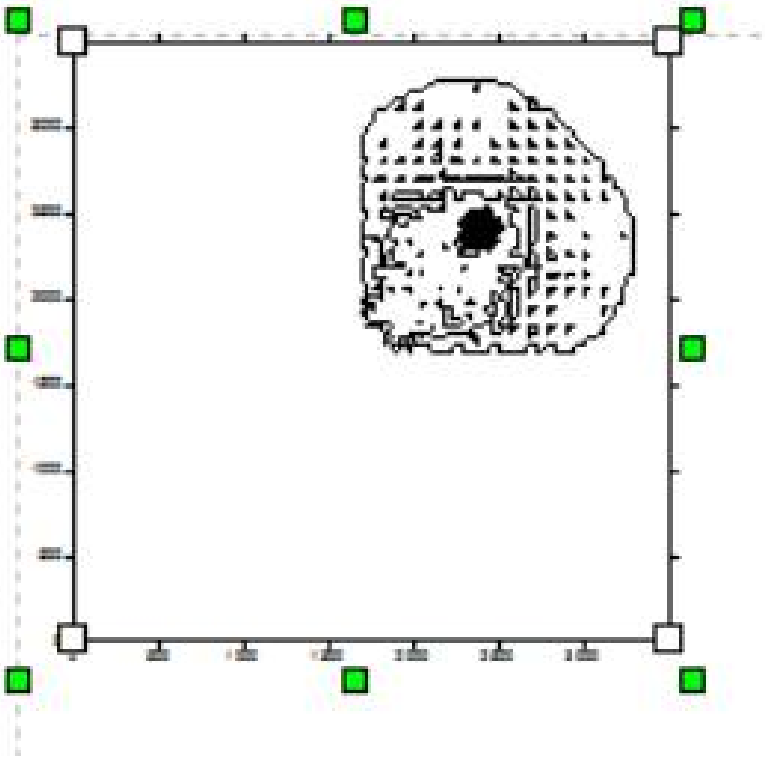

Рис. 5. Изменение коэффициента шероховатости 0,03 на глубине 10 м

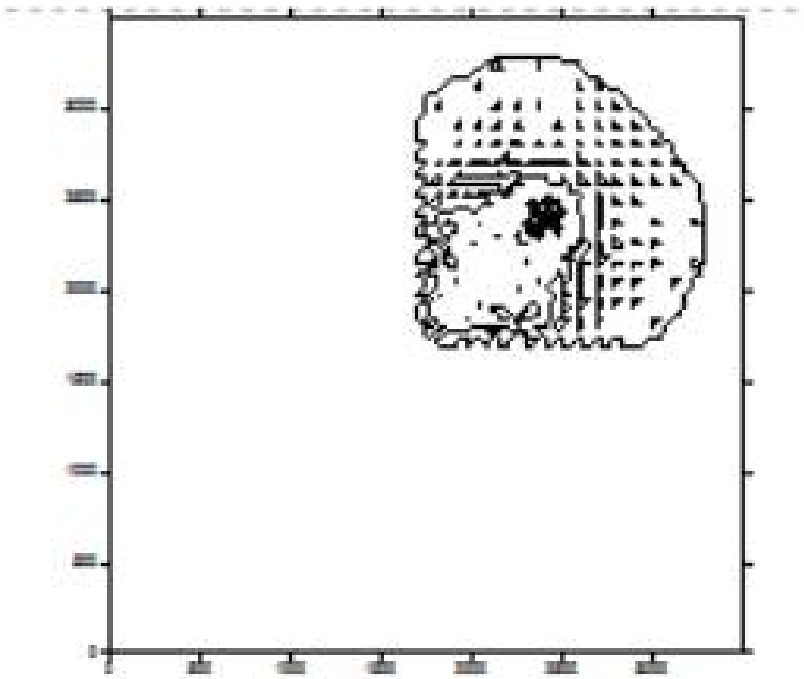

Рис. 6. Изменение коэффициента шероховатости 0,133 на глубине 10 м

На рис.7-8 изображено распространение загрязнения в зависимости от изменения коэффициента шероховатости ложа водного объекта n на глубине 1 метра. Можно сделать вывод, что изменение коэффициента шероховатости оказывает большее влияние на распространение загрязнения, при меньшей глубине. Чем больше коэффициент шероховатости, тем медленней происходит снижение концентрации загрязняющего вещества.

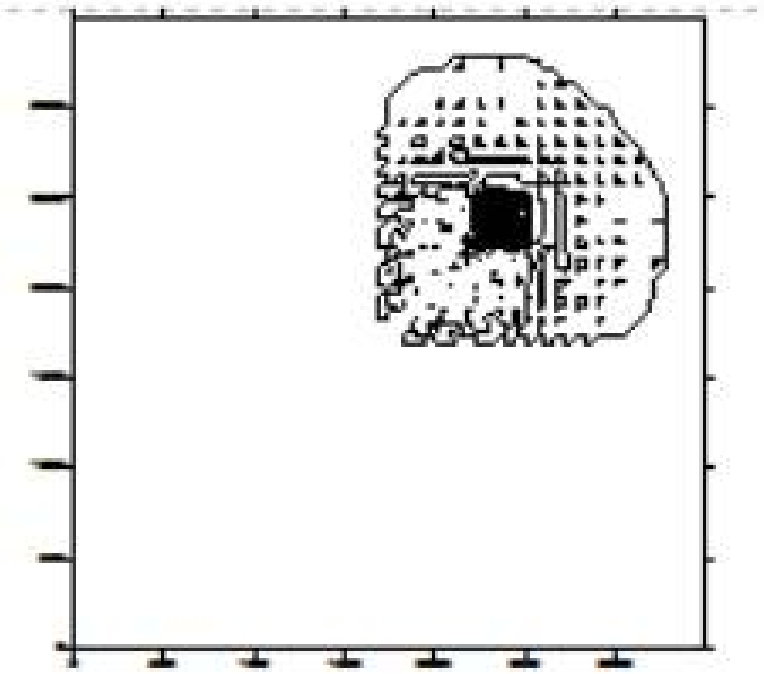

Рис. 7. Изменение коэффициента шероховатости 0,03 на глубине 1 м

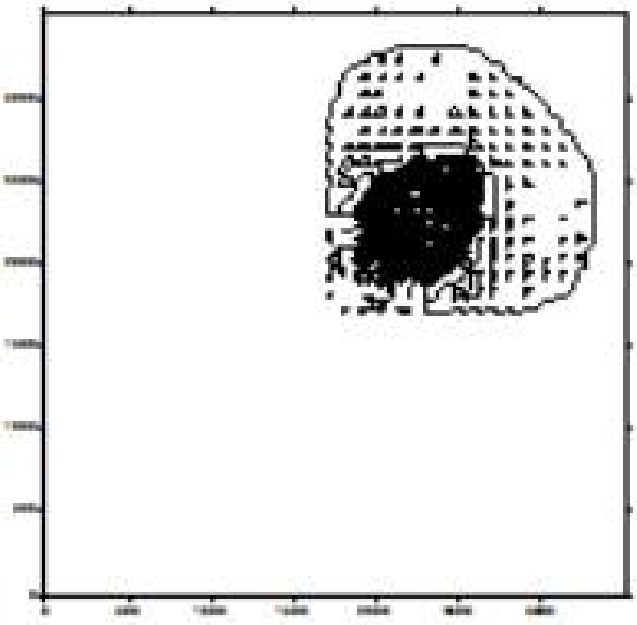

Рис. 8. Изменение коэффициента шероховатости 0,133 на глубине 1 м

На рис.9-13 изображено распространение загрязнения в зависимости от изменения средней глубины водного объекта. Так же как и коэффициент шероховатости, изменение средней глубины мало влияют на распространение загрязнения.

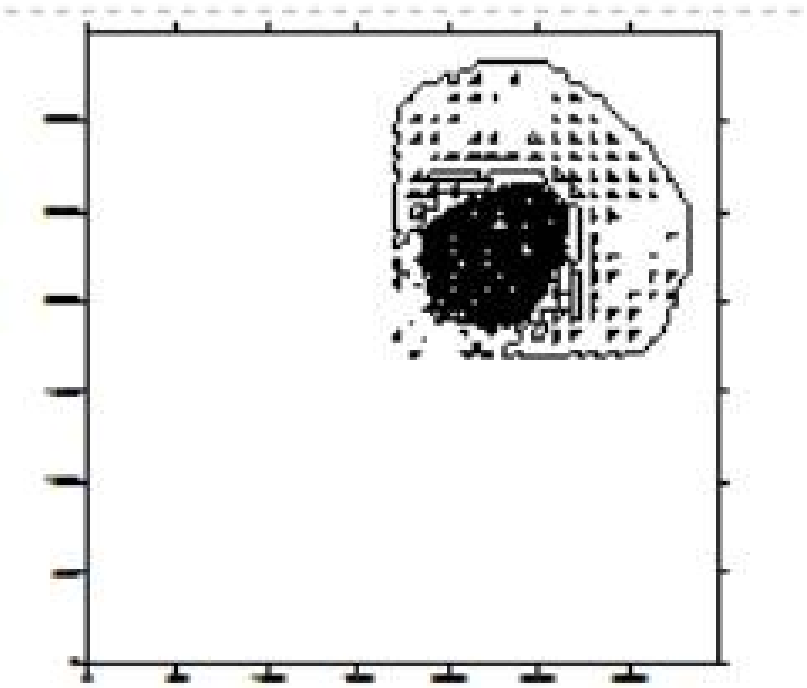

Рис.9.Изменение средней глубины водного объекта 0.5 м

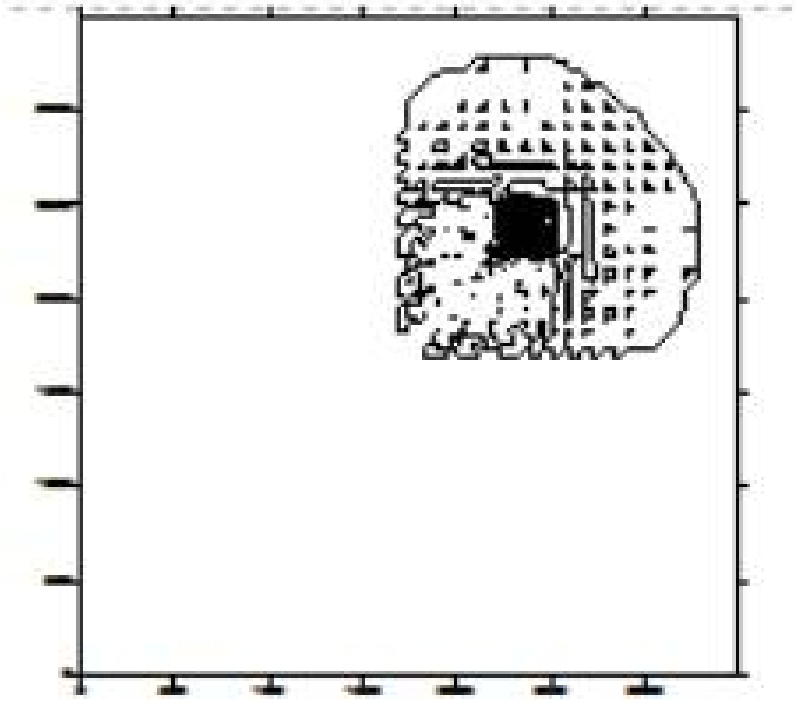

Рис.10.Изменение средней глубины водного объекта 1 м

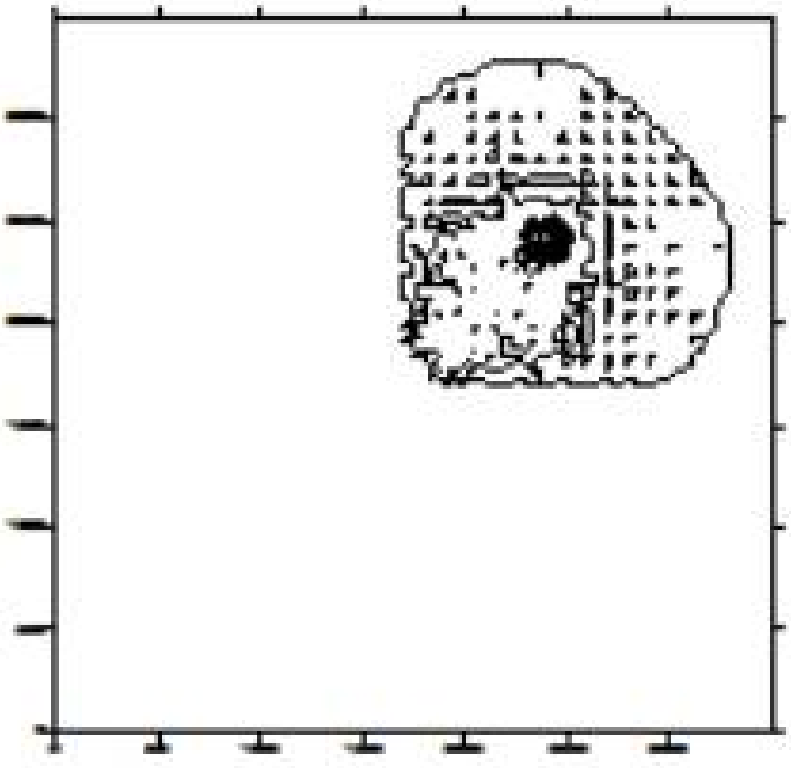

Рис.11.Изменение средней глубины водного объекта 3 м

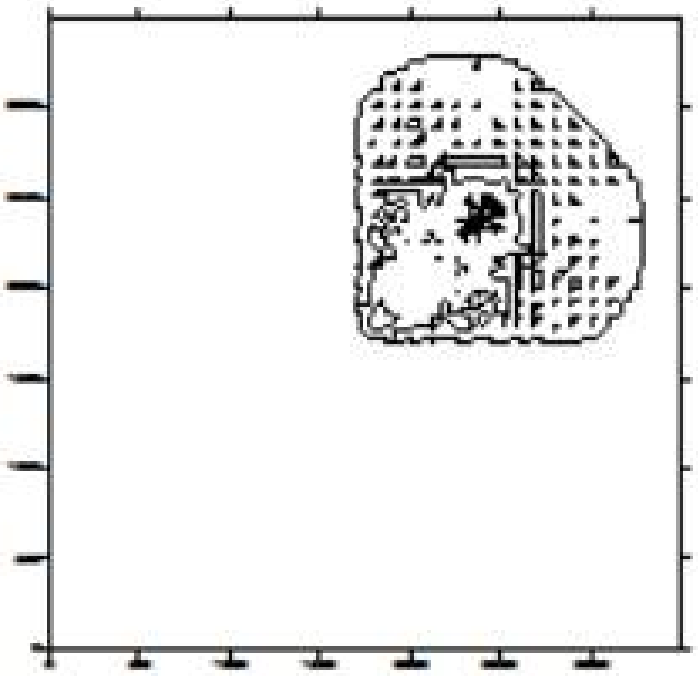

Рис.12.Изменение средней глубины водного объекта 10 м

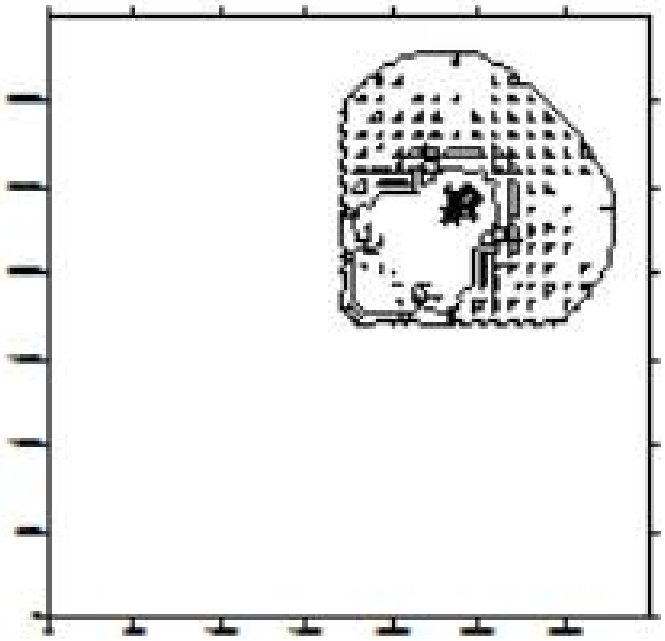

Рис.13.Изменение средней глубины водного объекта 100 м

На рисунках 14-16 показано распространение загрязнения через определенные промежутки времени.

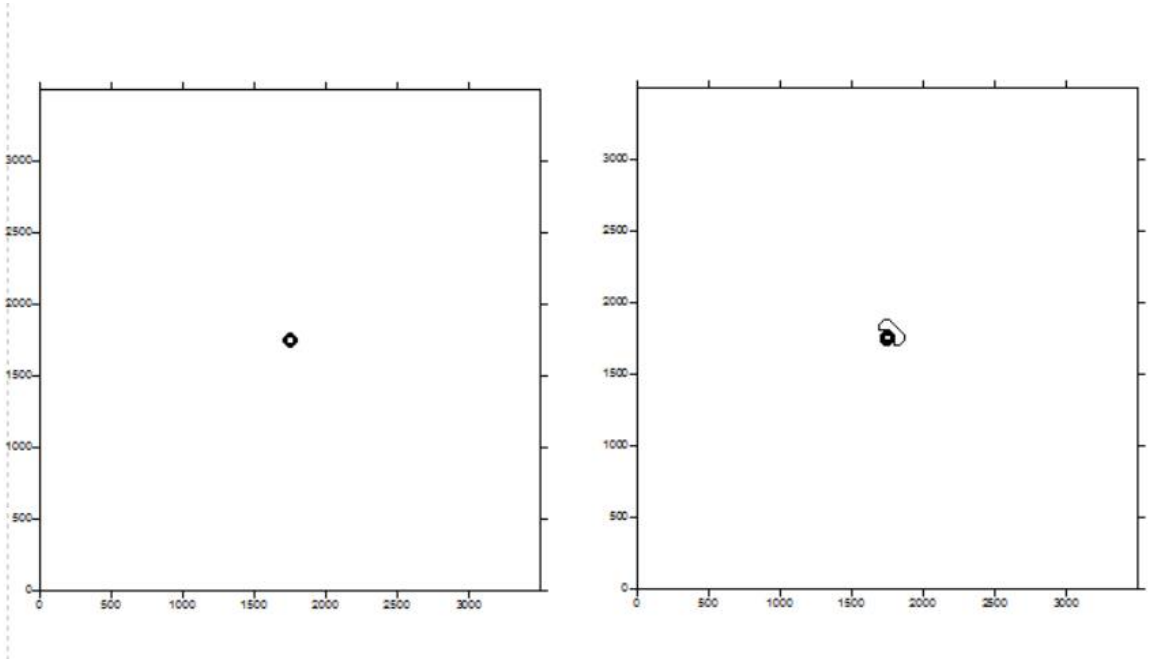

Рис.14. Распространение загрязнения спустя 1 с и 60 с.

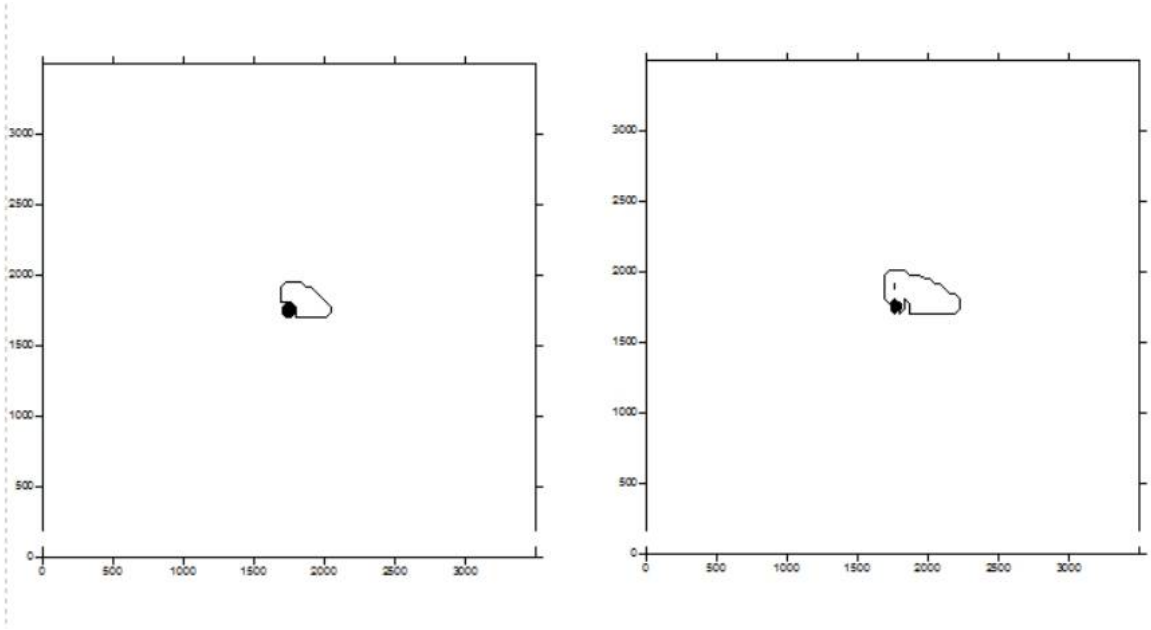

Рис.15. Распространение загрязнения спустя 10 мин. и 1 час.

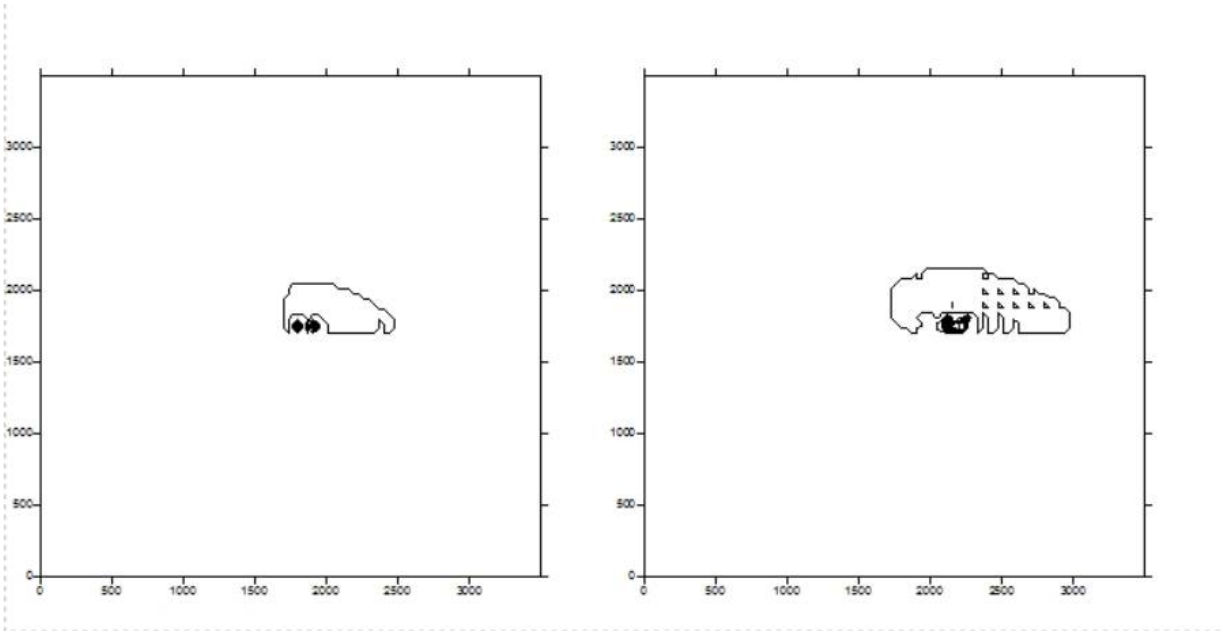

Рис 16. Распространение загрязнения спустя 3 часа и 9 часов

ЗАКЛЮЧЕНИЕ

В результате работы был разработан алгоритм вычисления концентрации вещества в каждый момент времени в любой точке сетки. Данный алгоритм был реализован с использованием технологий OpenMP и CUDA. Эта программа позволяет моделировать загрязнение озера каким-либо веществомидеструктуризацию этого вещества в любой момент времени.

Реализация алгоритма показала, что эта задача математического моделирования, хорошо подходит для решения с помощью видеокарты, так как использование GPU сокращает время исполнения программы примерно в 6 раз. Тест проводился далеко не на самой быстрой видеокарте по сегодняшним меркам – TeclaK20. Мне кажется, что если использовать самую последнюю видеокарту TeslaP100, то отставание процессора увеличится в десятки раз. При увеличении сетки отставание во времени для процессора будет еще больше. Выигрыш во времени достигается путем распараллеливания вычислений. Для каждой точки сетки должен производиться расчет концентрации вещества.

Данную программу можно распараллелить с использованием нескольких видеокарт на одном узле, а так же нескольких узлов с использованием технологий MPI.

Литература:

1.Мареев В. В., Станкова Е. Н. Основы методов конечных разностей. Санкт-Петербург: Издательство ВВМ, 2012.

2. Коэффициент турбулентной диффузии. Коэффициент Шези [Электронный ресурс]. URL: http://libraryno.ru/1-6-1-koefficient-turbulentnoy-diffuzii-koefficient-shezi-ekolog_audit/(дата обращения: 18.01.2017).

3. OpenMP и C++ [Электронный ресурс].URL: https://msdn.microsoft.com/ru-ru/library/dd335940.aspx(дата обращения: 20.01.2017).

4. Модели памяти CUDA [Электронный ресурс] URL: https://media.ls.urfu.ru/517/1341/3048/2938/1608/ (дата обращения 30.04.2017).

5. Введение в OpenMP: параллельное программирование на C++ [Электронный ресурс].URL: https://software.intel.com/ru-ru/blogs/2011/11/21/openmp-c(дата обращения: 20.01.2017).

6. Что такое OpenMP? [Электронный ресурс].URL: http://parallel.ru/tech/tech_dev/openmp.html (дата обращения: 20.01.2017).

7. Турбулентная диффузия [Электронный ресурс]URL:http://www.xumuk.ru/encyklopedia/2/4624.html(дата обращения: 26.01.2017).

8.Металлыисплавы.Справочник.Подред.Ю.П. Солнцева;НПОПрофессионал,НПОМирисемья;Санкт-Петербург, 2003 г.

9.Битюков Н.А., Исследование скоростного коэффициента в формуле Шези для открытых русел [Электронный ресурс]URL:http://www.rusnauka.com/13_EISN_2013/Geographia/3_136334.doc.html(дата обращения: 20.01.2017).

10. Гидродинамическое сопротивление [Электронный ресурс]. URL:http://bourabai.ru/physics/0771.html (дата обращения 22.03.2017).

11. Амосов А. А., Дубинский Ю .А., Копченова Н .В. Вычислительные методы для инженеров: Учеб.пособие. — М.: Высш. шк., 1994.

12. Метод конечных разностей [Электронный ресурс]. URL:http://www.stroitmeh.ru/lect83.htm (дата обращения 20.01.2017).

13.Меркулова Н. Н., Михайлов М. Д., Разностные схемы для обыкновенных дифференциальных уравнений: Учеб.пособие. – Томск: Национальный исследовательский томский государственный университет,2014.

14.Барышников Н. Б. Гидравлические сопротивления речных русел. - Санкт-Петербург: РГМУ, 2003.

15. Сиднев А. А., Сысоев А. В., Лекция №1. Обзор архитектуры современных многоядерных процессров: Учеб.пособие. – Нижний Новгород: Нижегородский государственный университет им. Н.И.Лобачевского, 2013.

16 Гергель В. П., Высокопроизводительные вычисления для многоядерных многопроцесных систем. Учеб.пособие. – Нижний Новгород: Нижегородский государственный университет им. Н.И.Лобачевского, 2010.

17.Сравнение центрального и графического процессора[Электронный ресурс]URL:https://tproger.ru/articles/cpu-and-gpu/ (дата обращения 30.04.2017).

18. Многопроцессорные системы. Классификация систем параллельной обработки данных.[Электронный ресурс]URL:http://citforum.ru/database/skbd/glava_10.shtml(дата обращения 17.02.2017)

19. Принцип векторной обработки. Векторно-конвейерные и матричные вычислительные системы. Факторы снижающие производительность векторных систем [Электронный ресурс]URL:http://www.hardline.ru/3/37/1479/(дата обращения 17.02.2017)

20.Лекция 3: Состояние и перспективы развития вычислительных систем и проектных технологий их создания [Электронный

ресурс]URL:http://www.intuit.ru/studies/courses/3456/698/lecture/14122?page=2 (дата обращения 15.02.2017)

21. Дацюк В.Н., Букатов А.А., Жегуло А.И. Электронное методическое пособие по курсу "Многопроцессорные системы и параллельное программирование" Часть I. Введение в организацию и методы программирования многопроцессорных вычислительных систем. Ростов-на-Дону, 2000.

22. Введение в проблематику разработки параллельных программ. [Электронный ресурс]URL:http://www.viva64.com/ru/a/0016/ (дата обращения 17.02.2017)

23. Таненбаум Э. Распределенные системы. Принципы и парадигмы. - СПб.: Питер, 2003.

24. Костенко В.А. К вопросу об оценке оптимальной степени параллелизма. М. Программирование. 1995.

25.Виды загрязнения водных ресурсов [Электронный ресурс]. URL: http://eco-garmony.blogspot.ru/2010/04/blog-post_07.html (дата обращения: 22.02.2017).

26. Марчук Г. И. Математическое моделирование в проблеме окружающей среды. М.: Наука, 1982. с. 23 – 45.

27. Математическое моделирование нефтяного загрязнения прибрежных вод республики Нигерия [Электронный ресурс]. URL: http://jurnal.org/articles/2008/ekol4.html(дата обращения: 22.02.2017).

28.Информационное моделирование загрязнения водных объектов [Электронный ресурс]. URL: http://www.izdatgeo.ru/pdf/gipr/2008-1/144.pdf(дата обращения: 22.02.2017).

29. Математическое моделирование динамики пассивной примеси в центральной части озера Селигер [Электронный ресурс]. URL: http://pmk-

vestnik.tversu.ru/issues/2007-4/19950136_2007_1_klimok.pdf(дата обращения: 22.02.2017).

30. Технология CUDAдля высокопроизводительных вычислений с использованием графических процессов [Электронный ресурс].URL: http://dvmh.keldysh.ru/attachments/download/345/cuda-n1.pdf(дата обращения: 20.01.2017).

Приложение1

Таблица П.1

Коэффициенты шероховатости водных объектов $n_ш$(таблица СкрибногоМ.Ф)

Значение $n_ш$	Условия
0,025	Естественные русла в весьма благоприятных условиях (чистые, прямые, незасоренные, земляные со свободным течением)
0,03	Русла постоянных водотоков равнинного типа, преимущественно больших и средних рек в благоприятных условиях состояниях ложа и течения воды Периодические водотоки (большие и малые) при очень хорошем состоянии поверхности и формы ложа
0,04	Сравнительно чистые русла постоянных равнинных водотоков в обычных условиях, извилистые, с некоторыми неправильностями в направлении струй или же прямые, но с неправильностями в рельефе дна (отмели, промоины, местами камни).
0,05	Русла больших и средних рек, значительно засоренные, извилистые и частично засоренные, каменистые, с неспокойным течением Периодические (ливневые и весенние) водотоки с крупногалечным или покрытым растительностью ложем Поймы больших и средних рек, сравнительно разработанные, покрытые растительностью (травой, кустарником)
0,067	Русла периодических водотоков сильно засоренные и извилистые

Значение $n_ш$	Условия
0,067	Поймы рек (промоины, кустарники, деревья с наличием заводей) Галечно-валунные русла горного типа с неправильной поверхностью водного зеркала Порожистые участки равнинных рек
0,08	Русла со слабым течением и поймы, значительно заросшие, с большими глубокими промоинами Валунные, горного типа русла с неправильной поверхностью водного зеркала (с летящими вверх брызгами воды)
0,1	Русла горно-водопадного типа с крупновалунным извилистым строением ложа, перепады ярко выражены, извилистость весьма сильная Поймы значительно заросшие, но с резко выраженным кососструйным течением, заводями и др.
0,133	Русла болотного типа (заросли, кочки, во многих местах почти стоячая вода и др.). Поймы с очень большими водными пространствами, с местными углублениями (озерами и др.)